Apache POI

- Creating and modifying files are common tasks you come across with.
- **Java** provides us different classes for File Manipulation.
- One of the file types we can manipulate are Excel files.
  - We can read and write on Excel file with the help of **Apache POI** library.

- To start, add below dependencies to the POM.XML

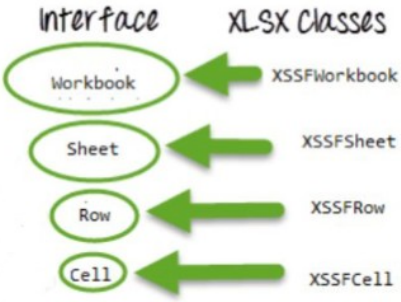
```
<dependency>
  <groupId>org.apache.poi</groupId>
  <artifactId>poi</artifactId>
  <version>4.1.0</version>
</dependency>
<dependency>
  <groupId>org.apache.poi</groupId>
  <artifactId>poi-ooxml</artifactId>
  <version>4.1.0</version>
</dependency>
```

- *Side Note: **Apache** is a most commonly used web-server.*
  - If Apache is mentioned by itself (without POI), it means to indicate Web Server.

Apache POI - Excel

Below are the Apache POI Interfaces and Classes we use to interact with Excel Files

- **XSSFWorkbook**: Is a class representation of **XLSX file**
  - This class implements **Workbook** interface
- **XSSFSheet**: Is a class representing a **sheet** in an XLSX file.
  - Implements Sheet interface
- **XSSFRow**: Is a class representing a **row** in the sheet of XLSX file
  - Implements Row interface
- **XSSFCell**: Is a class representing a **cell** in a row of XLSX file.
  - Implement Cell interface



Read Data From Excel File

- Create an object of FileInputStream class to read excel file
  - `FileInputStream fis = new FileInputStream("C:\\mydata.xlsx");`
- Create object of XSSFWorkbook class
  - `XSSFWorkbook workbook = new XSSFWorkbook(fis);`
- Create object of XSSFWorksheet class
  - `XSSFSheet sheet = workbook.getSheetAt(0);`
- Access a given row:
  - `XSSFRow row  = sheet .getRow(1);`
- Access a given cell:
  - `XSSFCell cell = row.getCell(1);`
  - `String value = cell.getStringCellValue();`

- Alternatively:
  - `String value = sheet .getRow(1).getCell(1).getStringCellValue();`

Get all the Data from Excel sheet

- To print all the data in the excel sheet, we need to use **nested loop**.
  - To get the number of rows present in sheet

```
int totalNoOfRows  = workSheet.getPhysicalNumberOfRows() ;
```

  - To get the number of columns present in sheet

```
int totalNoOfCols  = workSheet.getRow(i).getPhysicalNumberOfCells();
```

  - Use Nested Loop to print each cell

```
for (int i = 0; i <  totalNoOfRows; i++) {
            totalNoOfCols =  workSheet.getRow(i).getPhysicalNumberOfCells();
            for (int j = 0; j <  totalNoOfCols; j++) {

System.out.print(workSheet.getRow(i).getCell(j).getStringCellValue()+"\t");
            }
            System.out.println();
        }
```

Write Data to Excel File

- To write into Excel file, we identify the cell we want to write

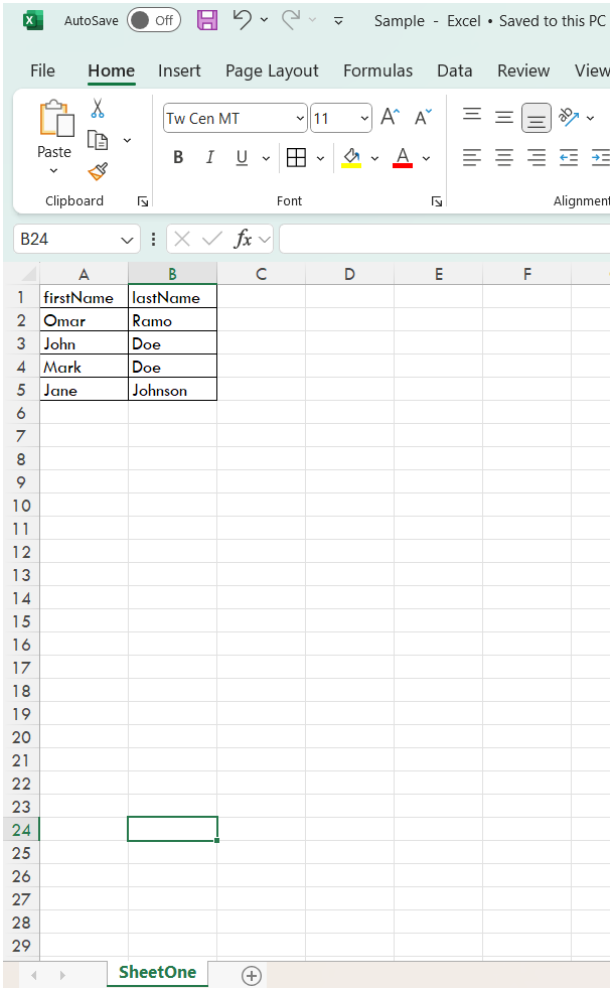
```
XSSFRow row = workSheet.getRow(RowNum);
if(row == null) {
    row = workSheet.createRow(RowNum);
}
XSSFCell cell = row.getCell(ColNum);
if (cell == null) {
   cell = row.createCell(ColNum);
}
cell.setCellValue(Result);
FileOutputStream fileOut = new FileOutputStream(path);
workBook.write(fileOut);
fileOut.flush();
fileOut.close();
```

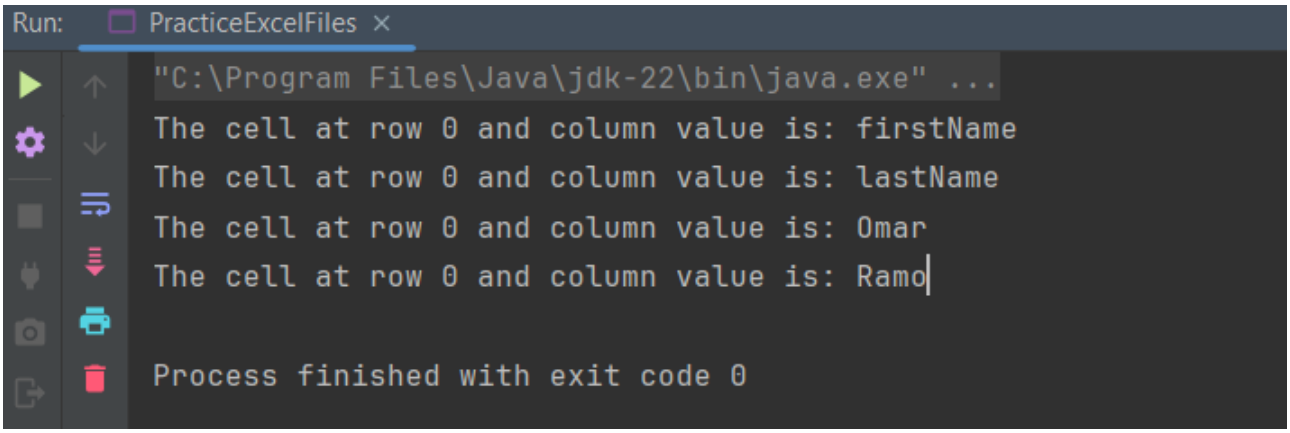
```
<dependency>
   <groupId>org.apache.poi</groupId>
   <artifactId>poi-ooxml</artifactId>
   <version>4.1.0</version>
</dependency>
<!-- https://mvnrepository.com/artifact/org.apache.poi/poi -->
<dependency>
   <groupId>org.apache.poi</groupId>
   <artifactId>poi</artifactId>
   <version>4.1.0</version>
</dependency>
```

HARD CODED WAY

```
package ui_automation.utilities;

import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.*;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;

public class PracticeExcelFiles {

    public static void main(String[] args) {


// We need to simply print the data from excel to our console
// The very first thing we need to have the path for the file we want to interact with
```

```java
// we can simply context click on the file itself and get the path
        String filePath = "src/test/resources/testData/Downloads/Sample.xlsx";
        try {
            FileInputStream contentOfTheFile = new FileInputStream(filePath);

// Once we have read the content of an excel file
// We can now start interacting with it
// 1. The first step we need to create a Workbook object
             Workbook workbook= null;
            try {
                 workbook = new XSSFWorkbook(contentOfTheFile);
             } catch (IOException e) {
                 throw new RuntimeException(e);
             }

// 2. Next, create a Sheet object that uses the workbook
// We can choose what sheet we want to use based on: index or name
// index
// Sheet sheetByIndex = workbook.getSheetAt(0);
// name
            //Sheet sheetByName = workbook.getSheet ( "sheetOne");
             Sheet sheetByName = workbook.getSheet("SheetOne");

// 3. Next, create an object of the Row
// We will utilize the sheet object and index to access a needed row
            Row row0 = sheetByName.getRow(0);
// 4. Finally, create an object of the cell
// We access it by row object and index of the cell
            Cell cell1 = row0.getCell(0);
            String cellValue1 = cell1.getStringCellValue();
            System.out.println("The cell at row 0 and column value is: " + cellValue1);
//Access to a different Data:
            Cell cell2 = row0.getCell(1);
            String cellValue2 = cell2.getStringCellValue();
            System.out.println("The cell at row 0 and column value is: " + cellValue2);

            //=========

            Row row1 = sheetByName.getRow(1);
// 4. Finally, create an object of the cell
// We access it by row object and index of the cell
            Cell cell3 = row1.getCell(0);
            String cellValue3 = cell3.getStringCellValue();
            System.out.println("The cell at row 0 and column value is: " + cellValue3);
//Access to a different Data:
            Cell cell4 = row1.getCell(1);
            String cellValue4 = cell4.getStringCellValue();
            System.out.println("The cell at row 0 and column value is: " + cellValue4);

        } catch (FileNotFoundException e) {
            throw new RuntimeException(e);


    }
}}
```




Run: PracticeExcelFiles
"C:\Program Files\Java\jdk-22\bin\java.exe" ...
The cell at row 0 and column value is: firstName
The cell at row 0 and column value is: lastName
The cell at row 0 and column value is: Omar
The cell at row 0 and column value is: Ramo
Process finished with exit code 0

DYNAMIC WAY OF ACCESSING ALL DATA

```
package ui_automation.utilities;

import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.*;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;

public class PracticeExcelFiles {
```

```
    public static void main(String[] args) {

        String filePath = "src/test/resources/testData/Downloads/Sample.xlsx";
          try {
              FileInputStream contentOfTheFile = new FileInputStream(filePath);


              Workbook workbook = null;
              try {
                  workbook = new XSSFWorkbook(contentOfTheFile);
              } catch (IOException e) {
                  throw new RuntimeException(e);
              }


           // DYNAMIC WAY OF ACCESSING ALL DATA
                      Sheet sheetByName = workbook.getSheet("SheetOne");
                      for (Row row : sheetByName) {
                          for (Cell cell : row) {
                              System.out.print(cell.getStringCellValue() + " | ");

                          }
                          System.out.println();
                      }
                  }
                      catch(FileNotFoundException e){
                          throw new RuntimeException(e);
                      }
                  }}
```

DYNAMIC WAY OF ACCESSING ALL DATA USING printAllDataFromExcelFile FROM ExcelUtility JAVA CLASS

```
public static void printAllDataFromExcelFile(String filePath, String SheetOne ) {
    try (FileInputStream contentOfTheFile = new FileInputStream(filePath)) {
        Workbook workbook = null;
        try {
            workbook = new XSSFWorkbook(contentOfTheFile);
        } catch (IOException ex) {
            throw new RuntimeException(ex);
        }
        Sheet sheetByName = workbook.getSheet("SheetOne");
        for (Row row : sheetByName) {
            for (Cell cell : row) {
                switch (cell.getCellType()) {
                    case STRING:
                        System.out.print(cell.getStringCellValue() + " | ");
                        break;
                    case NUMERIC:
                        System.out.print(cell.getNumericCellValue() + " | ");
                        break;
                    case BOOLEAN:
                        System.out.print(cell.getBooleanCellValue() + " | ");
                        break;
                }
            }
            System.out.println();
        }
        try {
            workbook.close();
        } catch (IOException ex) {
```

```
            throw new RuntimeException(ex);
        }
    } catch (IOException e) {
        e.printStackTrace();
        throw new RuntimeException("There was a problem reading the file at " + e);
    }

}
```

```
package ui_automation.utilities;

public class PracticeExcelFilesUtilitiy {
    public static void main(String[] args) {

        String filePath =
"src/test/resources/testData/Downloads/SampleDifDataTypes.xlsx";
        String sheetName="SheetOne";
    ExcelUtility.printAllDataFromExcelFile(filePath,sheetName);
    }
}
```

```
package ui_automation.utilities;

import org.apache.log4j.LogManager;
import org.apache.log4j.Logger;
import org.apache.poi.ss.usermodel.*;
import org.apache.poi.xssf.usermodel.XSSFCell;
import org.apache.poi.xssf.usermodel.XSSFRow;
import org.apache.poi.xssf.usermodel.XSSFSheet;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;

public class ExcelUtility {
    private static final Logger logger = LogManager.getLogger(ExcelUtility.class);
    private static XSSFWorkbook workBook;
    private static XSSFSheet workSheet;
    private static XSSFRow row;
    private static XSSFCell cell;

    //Reading a file
    public static void setExcelFile(String Path, String SheetName) throws Exception {
        try {
            FileInputStream inputStream = new FileInputStream(Path);
            workBook = new XSSFWorkbook(inputStream);
            workSheet = workBook.getSheet(SheetName);
            logger.info("Successfully connect to excel file!");
        } catch (Exception e) {
            logger.error("OOOOPPPSSS! Excel connection failed! Check your System!");
            throw (e);
        }
    }

    //Converting Data Type : (Look to the TWO IMPROVED METHODS for converting
Data Type:printAllDataFromExcelFile AND getCellDataImproved)
    public static Object getCellData(int RowNum, int ColNum) throws Exception {
```

```
        try {
            cell = workSheet.getRow(RowNum).getCell(ColNum);
            return cell.getStringCellValue();
        } catch (Exception IllegalStateException) {
            try {
                cell = workSheet.getRow(RowNum).getCell(ColNum);
                return cell.getNumericCellValue();
            } catch (Exception e) {
                cell = workSheet.getRow(RowNum).getCell(ColNum);
                return cell.getDateCellValue();
            }
        }
    }

    public static String getCellDataAsString(int RowNum, int ColNum) throws Exception {
        try {
            cell = workSheet.getRow(RowNum).getCell(ColNum);
            return cell.getStringCellValue();
        } catch (Exception e) {
            throw (e);
        }
    }

    public static double getCellDataAsDouble(int RowNum, int ColNum) throws Exception {
        try {
            cell = workSheet.getRow(RowNum).getCell(ColNum);
            return cell.getNumericCellValue();
        } catch (Exception e) {
            throw (e);
        }
    }

    public static void setCellData(String path, String value, int RowNum, int ColNum)
throws Exception {
        try {
            row = workSheet.getRow(RowNum);
            cell = row.getCell(ColNum);
            if (cell == null) {
                cell = row.createCell(ColNum);
                cell.setCellValue(value);
            } else {
                cell.setCellValue(value);
            }
            FileOutputStream fileOut = new FileOutputStream(path);
            workBook.write(fileOut);
            fileOut.flush();
            fileOut.close();
        } catch (Exception e) {

            throw (e);

        }

    }


    public static void createExcelAndWrite(String fileName, String value) {
        workBook = new XSSFWorkbook();
        workSheet = workBook.createSheet("FIRST SHEET");
        row = workSheet.createRow(0);
        cell = row.createCell(0);
        cell.setCellValue(value);
```

```
        try (FileOutputStream fos = new FileOutputStream(new File(fileName))) {
            workBook.write(fos);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
//CONVERTING DATA TYPES
    public static void printAllDataFromExcelFile(String filePath, String SheetOne) {
        try (FileInputStream contentOfTheFile = new FileInputStream(filePath)) {
            Workbook workbook = null;
            try {
                workbook = new XSSFWorkbook(contentOfTheFile);
            } catch (IOException ex) {
                throw new RuntimeException(ex);
            }
            Sheet sheetByName = workbook.getSheet("SheetOne");
            for (Row row : sheetByName) {
                for (Cell cell : row) {
                    switch (cell.getCellType()) {
                        case STRING:
                            System.out.print(cell.getStringCellValue() + " | ");
                            break;
                        case NUMERIC:
                            System.out.print(cell.getNumericCellValue() + " | ");
                            break;
                        case BOOLEAN:
                            System.out.print(cell.getBooleanCellValue() + " | ");
                            break;
                    }
                }
                System.out.println();
            }
            try {
                workbook.close();
            } catch (IOException ex) {
                throw new RuntimeException(ex);
            }
        } catch (IOException e) {
            e.printStackTrace();
            throw new RuntimeException("There was a problem reading the file at " + e);
        }

    }


    // Improved version of getCellData
    public static Object getCellDataImproved(int rowNum, int colNum) {
        try {
            cell = workSheet.getRow(rowNum).getCell(colNum);
            if (cell == null) {
// Return a default value or indicate that the cell is empty/blank
                return "";
                //  or you can return null;
            }
            switch (cell.getCellType()) {

                case STRING:
                    return cell.getStringCellValue();
                case NUMERIC:

                    if (DateUtil.isCellDateFormatted(cell)) {
```

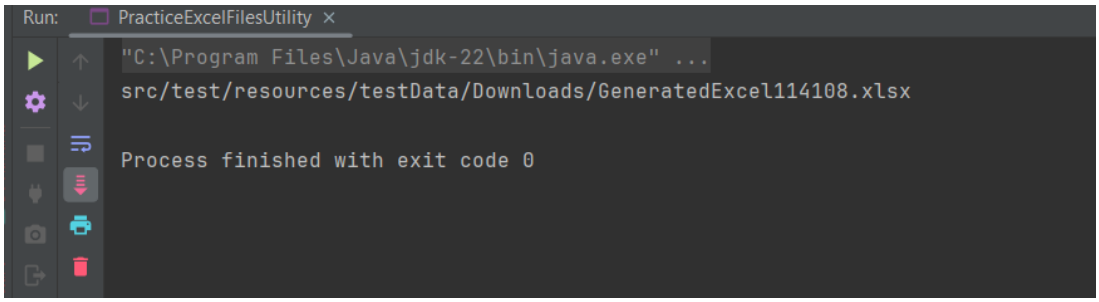
```
                    return cell.getDateCellValue();
                } else {

                    return cell.getNumericCellValue();
                }
            case BOOLEAN:
                return cell.getBooleanCellValue();
            case FORMULA:
// Handle formula cells if necessary
                return cell.getCellFormula();
            case BLANK:
                return "";

            // or handle blank cells as needed
            default:
                return "Unsupported cell type:" + cell.getCellType();
        }
    } catch (IllegalStateException e) {
// Log the exception or handle it as per your application's requirement
        e.printStackTrace();
        return "Error reading cell data";
    }

}


/**
 * Creates a new Excel file with a single sheet.
 *
 * @param filePath  The file path where the new Excel file will be created.
 * @param sheetName The name of the new sheet in the Excel file.
 */
public static void createNewExcelFile(String filePath, String sheetName) {
// Create a workbook object. XSSFWorkbook is for .xlsx files.
    Workbook workbook = new XSSFWorkbook();
// Create a sheet in the workbook
    Sheet sheet = workbook.createSheet(sheetName);
    try (FileOutputStream fileOut = new FileOutputStream(filePath)) {
// Write the workbook with the new sheet to the file
        workbook.write(fileOut);
        System.out.println("New Excel file with a sheet named" + sheetName + "
'created at:" + filePath);
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
// Close the workbook
        try {
            workbook.close();
        } catch (IOException e) {
            e.printStackTrace();


        }

    }

}
/**
 * Writes data to a specific cell in an existing or new Excel file and sheet.
 *
```

```
     * @param filePath  The file path of the Excel file.
     * @param sheetName The name of the sheet.
     * @param rowNum    The row number of the cell (0-based).
     * @param colNum    The column number of the cell (0-based).
     * @param data      The data to be written to the cell.
     */
    public static void writeToCell(String filePath, String sheetName, int rowNum, int
colNum, String data) {
        FileInputStream fileIn = null;
        FileOutputStream fileOut = null;
        Workbook workbook = null;
        try {
            fileIn = new FileInputStream(new File(filePath));
            workbook = new XSSFWorkbook(fileIn);
            Sheet sheet = workbook.getSheet(sheetName);
// If the sheet doesn't exist, create it.
            if (sheet == null) {
                sheet= workbook.createSheet(sheetName);}
// Get the row or create it if it doesn't exist
                Row row = sheet.getRow(rowNum);
                if (row == null) {
                    row = sheet.createRow(rowNum);
                                }
                // Get the cell or create it if it doesn't exist
                Cell cell = row.getCell(colNum);
                if (cell == null) {
                    cell= row.createCell(colNum);}
// Set the cell value
                    cell.setCellValue(data);
// Write the changes back to the file
                    fileOut = new FileOutputStream(new File(filePath));
                    workbook.write(fileOut);
                } catch (IOException e) {
                    e.printStackTrace();
                } finally{
// Close resources
                    try {
                        if (workbook != null) {

                        workbook.close();}
                        if (fileIn != null) {
                            fileIn.close();
                        }
                        if (fileOut != null) {
                            fileOut.close();
                        }
                    } catch (IOException e) {
                        e.printStackTrace();
                    }
                }
            }


}
```

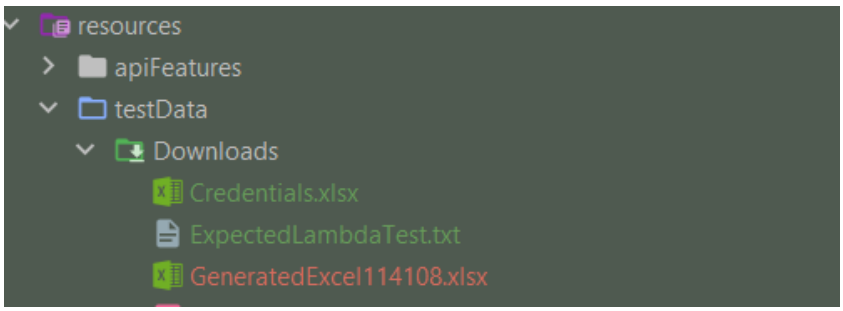
CREATE AND WRITE DATA TO EXCEL FILE

```java
package ui_automation.utilities;

import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;

import java.io.FileOutputStream;
import java.io.IOException;
import java.time.LocalTime;
import java.time.format.DateTimeFormatter;

public class PracticeExcelFilesUtility {
    public static void main(String[] args) {
// CREATE AND WRITE DATA TO EXCEL FILE
        String time = LocalTime.now().format(DateTimeFormatter.ofPattern("HHmmss"));
        String fileName = "GeneratedExcel " + time + ".xlsx";
//Time
        System.out.println(fileName);
        String filePath = "src/test/resources/testData/Downloads/" + fileName;
        System.out.println(filePath);
// first create the FileOutputStream and the workbook
        try {
            Workbook workbook = new XSSFWorkbook();
            FileOutputStream fileOut = new FileOutputStream(filePath);

// 1. Create the sheet in the workbook
            Sheet sheet = workbook.createSheet("newSheet");
// 2. Create a Row in the sheet
            Row row1 = sheet.createRow(0);
// 3. Create cells to that row
            Cell cell1 = row1.createCell(0);
            cell1.setCellValue("ID");
            Cell cell2 = row1.createCell(1);
            cell2.setCellValue("Full Name");
// Create a new row and populate it
            Row row2 = sheet.createRow(1);
// short hand of creating cell object inline method
            row2.createCell(0).setCellValue(1001);
            row2.createCell(1).setCellValue("Omar Ramo");

// resize the columns based on content
            for (int i = 0; i < row1.getPhysicalNumberOfCells(); i++) {

                sheet.autoSizeColumn(i);
            }
// write the data to workbook
            workbook.write(fileOut);
            fileOut.flush();

        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

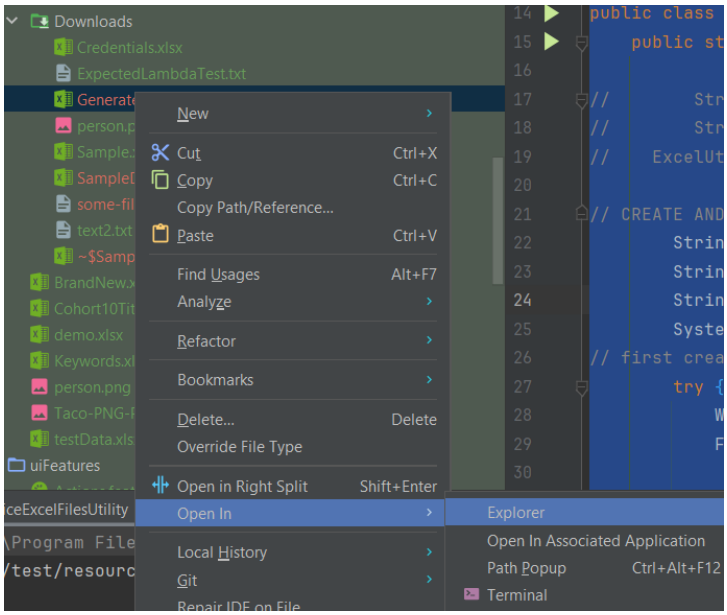
```
}
```



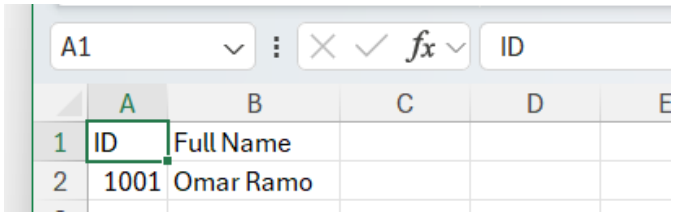
The file is created under Downloads:



To open it:

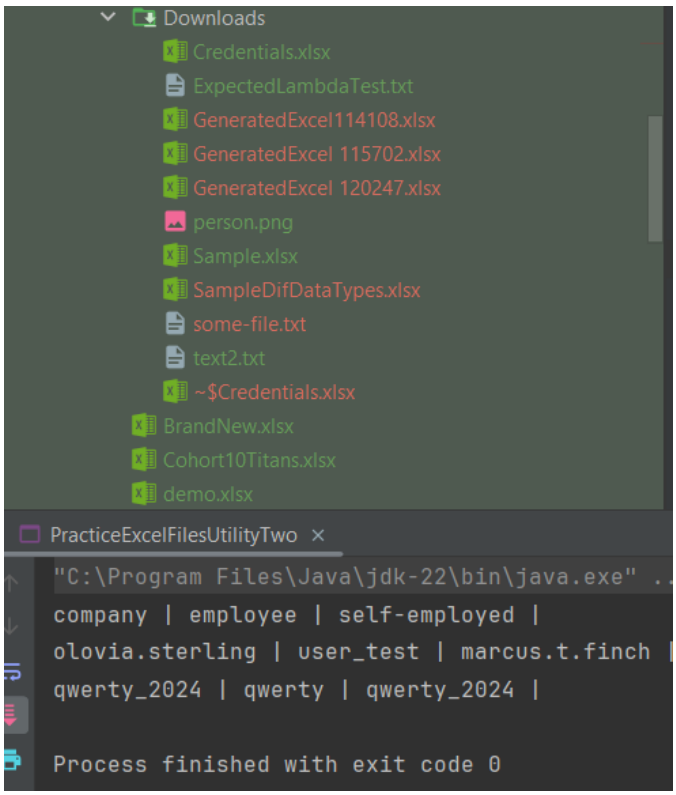


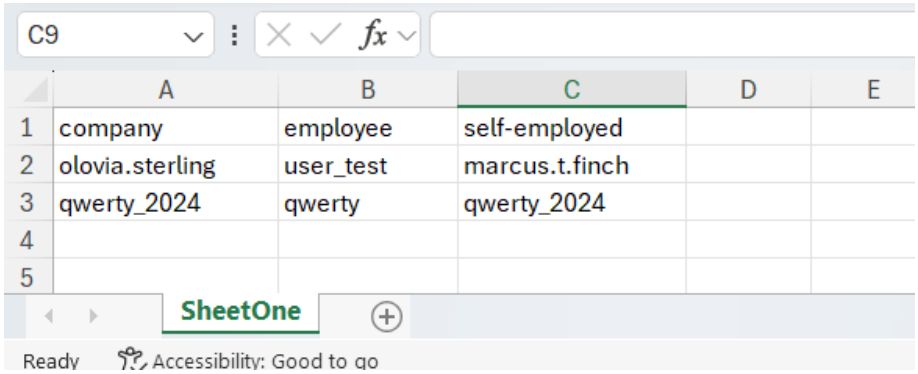
// Resize the columns based on content



//Print all Data:

```
package ui_automation.utilities;

public class PracticeExcelFilesUtilityTwo {
    public static void main(String[] args) {

    String filePath = "src/test/resources/testData/Downloads/Credentials.xlsx";
       //Print all Data:
    String sheetName="credentials";
   ExcelUtility.printAllDataFromExcelFile(filePath,sheetName);

}}
```

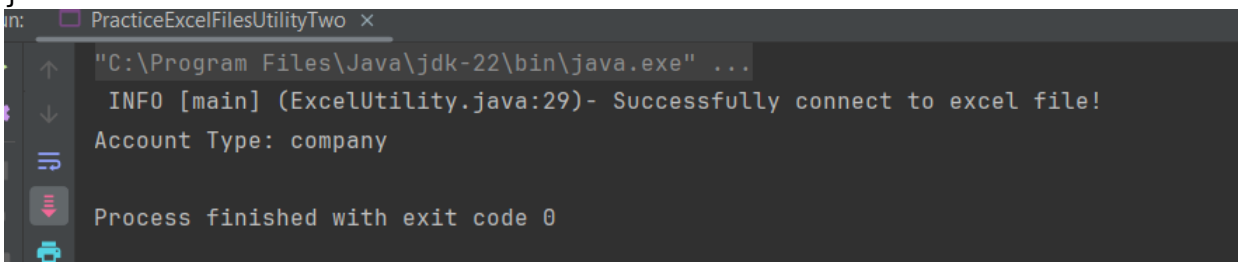




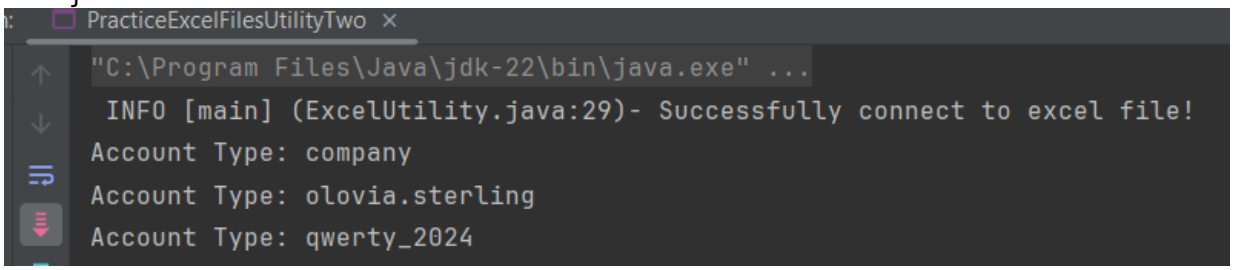
//Print Data of Cell:(0,0):

```
// Set the file for ExcelUtilities to have it in the JVM
        try {
            ExcelUtility.setExcelFile(filePath, sheetName);
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
```

```
        String accountType = null;
        try {
            accountType = ExcelUtility.getCellDataImproved(  0,  0).toString();
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
        System.out.println("Account Type: " + accountType);
}
```


PracticeExcelFilesUtilityTwo
"C:\Program Files\Java\jdk-22\bin\java.exe" ...
INFO [main] (ExcelUtility.java:29)- Successfully connect to excel file!
Account Type: company
Process finished with exit code 0

```
try {
            ExcelUtility.setExcelFile(filePath, sheetName);
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
        String accountType = null;
        try {
            accountType = ExcelUtility.getCellDataImproved(  0,  0).toString();
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
        System.out.println("Account Type: " + accountType);

        String username = ExcelUtility.getCellDataImproved(  1,  0).toString();
        System.out.println("Account Type: " + username);

        String password = ExcelUtility.getCellDataImproved(  2,  0).toString();
        System.out.println("Account Type: " + password);
    }
```

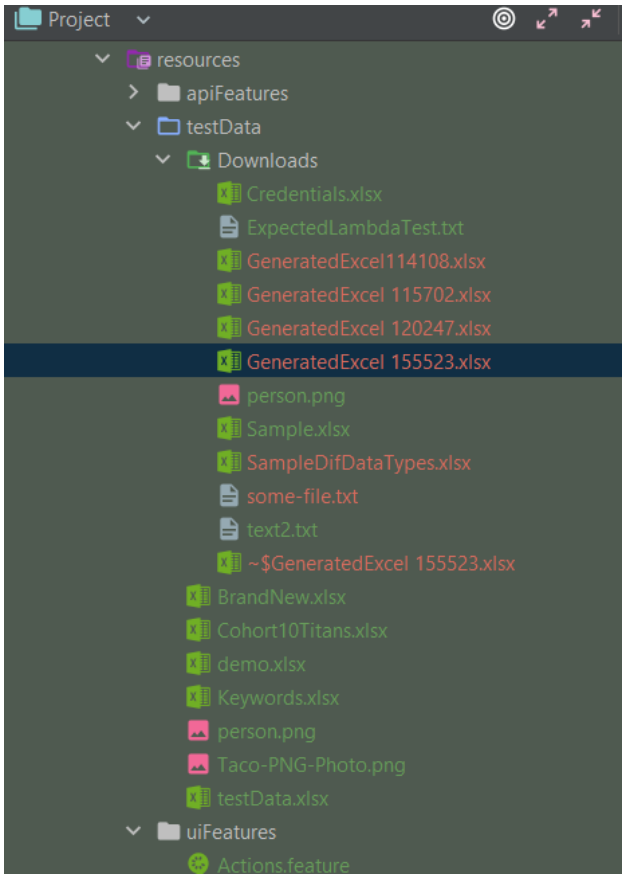


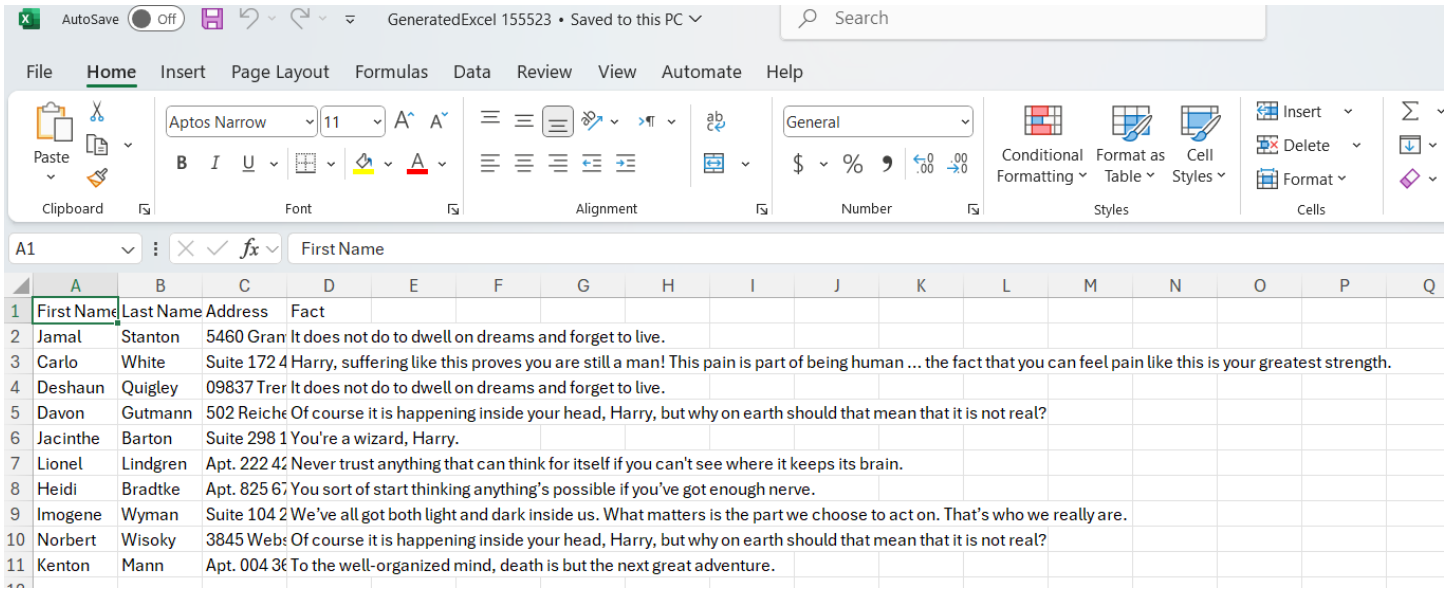
Creating a new Excel file, and Writing with Fake Data : checknuris

```
package ui_automation.utilities;

import com.github.javafaker.Faker;

import java.time.LocalTime;
import java.time.format.DateTimeFormatter;

public class PracticeExcelUtilityThree {
    public static void main(String[] args) {
        String time = LocalTime.now().format(DateTimeFormatter.ofPattern("HHmmss"));
        String fileName = "GeneratedExcel " + time + ".xlsx";
```

```
        String filePath = "src/test/resources/testData/Downloads/"+fileName;
        String sheetName = "NewSheet";

         //Creating a new Excel File :
        ExcelUtility.createNewExcelFile(filePath, sheetName);
         //Writing on the new Excel File :
        ExcelUtility.writeToCell(filePath, sheetName,0,0,"First Name");
        ExcelUtility.writeToCell(filePath, sheetName,0,1,"Last Name");
        ExcelUtility.writeToCell(filePath, sheetName,0,2,"Address");
        ExcelUtility.writeToCell(filePath, sheetName,0,3,"Fact");
         //Using Java Faker to populate the cells
        //pom.xml should contains:

//        <dependency>
//            <groupId>com.github.javafaker</groupId>
//            <artifactId>javafaker</artifactId>
//            <version>0.16</version>
//        </dependency>

        Faker faker = new Faker();
        for (int i = 1; i <= 10; i++) {
            for (int j = 0; j < 4; j++){
                switch (j) {
                    case 0:
                        ExcelUtility.writeToCell(filePath, sheetName, i, j,
faker.name().firstName());
                        break;
                    case 1:
                        ExcelUtility.writeToCell (filePath, sheetName, i, j,
faker.name().lastName());
                        break;
                    case 2:
                        ExcelUtility.writeToCell (filePath, sheetName, i, j,
faker.address().fullAddress());
                        break;
                    case 3:
                        ExcelUtility.writeToCell (filePath, sheetName, i, j,
faker.harryPotter().quote());
                        break;
                }
    }
}}}
```

```
"C:\Program Files\Java\jdk-22\bin\java.exe" ...
New Excel file with a sheet namedNewSheet 'created at:src/test/resources/testData/Downloads/GeneratedExcel 155523.xlsx


Process finished with exit code 0
```





	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	First Name	Last Name	Address	Fact													
2	Jamal	Stanton	5460 Gran	It does not do to dwell on dreams and forget to live.													
3	Carlo	White	Suite 172 4	Harry, suffering like this proves you are still a man! This pain is part of being human ... the fact that you can feel pain like this is your greatest strength.													
4	Deshaun	Quigley	09837 Trer	It does not do to dwell on dreams and forget to live.													
5	Davon	Gutmann	502 Reiche	Of course it is happening inside your head, Harry, but why on earth should that mean that it is not real?													
6	Jacinthe	Barton	Suite 298 1	You're a wizard, Harry.													
7	Lionel	Lindgren	Apt. 222 42	Never trust anything that can think for itself if you can't see where it keeps its brain.													
8	Heidi	Bradtke	Apt. 825 67	You sort of start thinking anything's possible if you've got enough nerve.													
9	Imogene	Wyman	Suite 104 2	We've all got both light and dark inside us. What matters is the part we choose to act on. That's who we really are.													
10	Norbert	Wisoky	3845 Webs	Of course it is happening inside your head, Harry, but why on earth should that mean that it is not real?													
11	Kenton	Mann	Apt. 004 36	To the well-organized mind, death is but the next great adventure.													